

JavaScript Cheatsheet – Admincamp 2012 Hands-On 6

<p><u>Hello World:</u></p> <pre><script type="text/javascript"> //message box Hello World alert("Hello World"); //schreibt Hallo Welt auf die Webseite document.write('<p>Hallo Welt</p>'); </script></pre>	<p><u>Typen:</u></p> <p>String: Immer in Anführungszeichen</p> <p>object: (auch z.B. arrays)</p> <p>null: Eigenschaften ohne Wert bzw. Funktionen ohne Rückgabewert</p> <p>undefined: nicht definierte oder gelöschte Variable</p>
<p><u>Funktionen:</u></p> <pre>function Funktionsname(parameter) { //Anweisung(en) //optionale Rückgabe, beendet die Ausführung: return "Hello World"; }</pre> <p>Rückgabebetyp nicht definiert; Parameter nicht typisiert;</p>	<p><u>Number:</u></p> <p>64-bit Gleitkommazahlen („double“)</p> <p>//Ganzzahl</p> <pre>var number = parseInt(Stringvalue); //Gleit"komma"zahl, wenn Dezimalpunkt vorhanden var number = parseFloat(Stringvalue);</pre> <p>NaN = Not a Number, „ansteckend“</p> <p>-> Test mit: isNaN(Variablen-Name)</p>
<p><u>Kommentare:</u></p> <pre>//Einzeiliger Kommentar /* Mehrzeiliger Kommentar*/</pre>	<p><u>Boolean:</u></p> <p>Falsche Werte: false, „null“, „undefined“, "" (leere Zeichenkette), 0, NaN</p> <p>Alles andere ist „true“ - auch „0“ oder „false“, da gültige Strings</p>
<p><u>Variablen:</u></p> <pre>//scope "Global" variablenNameGlobal = Wert; //scope Lokal = Aktuelle Funktion/Objekt var variablenNameLokal = Wert;</pre>	<p><u>Array:</u></p> <pre>//festes Array var arrayFix = new Array(100); var arrayFix = [100]; //Kurzform //dynamisches Array var arrayDyn = new Array(); //wenn Werte bereits bekannt sind var arrayBek = new Array("Eins", "Zwo");</pre>
<p><u>Bedingte Anweisung:</u></p> <pre>if(Bedingung) { //Block von n Anweisungen } else if (andereBedingung) TuWasAnderes(); //EineAnweisung else DannHaltDas();</pre>	<pre>alert(arrayBekannt[0]); // "Eins" arrayBekannt[0] = "Vier"; //Zuweisung</pre>

<u>Vergleiche:</u> <ul style="list-style-type: none"> • <code>==</code> und <code>!=</code> vergleichen Werte • <code>===</code> und <code>!==</code> vergleichen Werte unter Berücksichtigung des Typs • <code><</code>, <code><=</code>, <code>></code>, <code>>=</code>, <code>!</code> usw. wie in LotusScript 	<u>Logische Operatoren:</u> <p>"Ausdruck1 && Ausdruck2" gibt Ausdruck1 aus, wenn Ausdruck1 falsch ist, sonst Ausdruck2</p> <p>"Ausdruck1 Ausdruck2" gibt Ausdruck1 aus, wenn wahr, sonst Ausdruck2</p>
<u>Schleifen – For:</u> <pre>for([init];[Bedingung];[Increment]){ Anweisungen} - break; unterbricht die Ausführung - continue; direkt zum nächsten Increment</pre>	<u>Schleifen – While:</u> <pre>while(Bedingung){ Anweisungen(en); //ggf. ein Increment } - oder: do{ Anweisungen(en); //ggf. ein Increment }while(Bedingung)</pre>
<u>Elemente und Werte:</u> <pre>document.getElementById(id); //Liefert alle Elemente dieses Namens document.getElementsByName(name); //gibt das "value"-Attribut zurück var variable = Element.value</pre>	<u>Attribute und Styles:</u> <pre>//(er)setzt den Attributwert element.setAttribute(Name,Wert); //direkter Zugriff auf den Style element.style.styleName = styleWert;</pre>
<u>Events:</u> <pre><!-- Event als Attributknoten -> <input type="button" value="Label" onclick="Anweisungen();" /> //Event als Objekteigenschaft element.onclick = Funktion();</pre>	<u>Sichtbarkeit:</u> <pre>//unsichtbar style="display:none;" //sichtbar style="display:block;"</pre>